CLAIMS

1. A computer readable medium comprising computer-program instructions executable by a processor and implementing instructions for:

a runtime hosting interface comprising a host abstraction interface (HAI), the HAI corresponding to execution environment abstraction(s) supported by a host application, at least one specific interface or object corresponding to a specific one HAI being accessible by a runtime during execution of runtime managed code and responsive to an action or event associated with an identified one HAI, the HAI providing an interface for the runtime to configure host execution environment parameters and/or notify the host of a runtime event.

2. A computer-readable medium as recited in claim 1, wherein an interface of the HAI provides the runtime with a pointer to an object associated with the interface, the object for calling by the runtime responsive to a specified event or criteria.

3. A computer-readable medium as recited in claim 1, wherein the one or more execution environment abstractions correspond to management services for memory, threads/tasks, I/O completion, synchronization, runtime entry/exit notification, security context, impersonation, runtime configuration, customized assembly loading, host protection, garbage collection (GC), debugging, and/or executable code service abstractions.

4. A computer-readable medium as recited in claim 1, wherein the runtime hosting interface further comprises a runtime interface (RI) for use by the host application to configure operations of the runtime, notify the runtime of an event, and/or to obtain additional information during host application process execution.

5. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to identify host application implemented ones of the HAI or associated object(s) for subsequent calling responsive to an action or event associated with an identified one of the respective execution environment abstractions.

6. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to request the host application to perform a memory allocation.

7. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to communicate a low memory notification from an OS to the host application, and/or inform the host application of consequences of failing a particular memory allocation via an HAI.

8. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to create a new thread/task via the HAI.

9. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to reuse or pool a runtime-implemented portion of a task via the HAI.

10. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to notify the host application that a task cannot be moved to a different physical OS thread and cannot have execution of the task blocked for a specified window of time.

11. A computer-readable medium as recited in claim 1, wherein the HAI comprises:

an interface for the runtime to indicate a callback to the host application, the callback for notifying the runtime when a task has been moved to a runnable or non-runnable state; and

if the task has been moved to a non-runnable state, an interface to specify that the task is to be rescheduled as soon as possible by the host application.

12. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to:

queue a thread/task to a host application implemented thread pool;

set a size of the host application implemented thread pool; and/or

query the host application implemented thread pool.

13. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to:

provide a callback to the host application for notifying the runtime that a task has been moved to a different locale or a locale has changed for the task; and/or

notify the host application, that a task has been moved to a different locale or a locale has changed for the task.

14. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to delay the host application abort of a task.

15. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to modify an object identified by an interface of the HAI.

16. A computing device as recited in claim 15, wherein the object is a task priority.

17. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to notify the host application that a task/thread is to leave the runtime into unmanaged code.

18. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to notify the host application that a task/thread is to reverse-leave the runtime into unmanaged code.

19. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to notify the host application that a task/thread is to enter the runtime from unmanaged code.

20. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to notify the host application that a task/thread is to reverse-enter the runtime from unmanaged code.

21. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to specify a maximum number of threads/tasks that will be available to service requests on one or more I/O completion ports.

22. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to binding a handle to an I/O completion port of the host application.

23. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to supply the host application with a runtime implemented callback, the runtime implemented callback for invoking by the host application when an asynchronous I/O operation completes.

24. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to supply a runtime implemented callback to the host application, the runtime implemented callback to be invoked by the host application when an asynchronous I/O operation completes, the runtime implemented callback being used by the runtime to provide custom state information to the host application.

25. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to:

generate a task; and

specify one or more synchronization objects for the task to substantially ensure host application knowledge of a lock on the task, the one or more synchronization objects comprising a critical section, a manual and/or auto-reset event, a semaphore, a reader/writer lock, and/or a task monitor.

26. A computer-readable medium as recited in claim 1, wherein the HAI comprises an interface for the runtime to notify the host application of one or more runtime interfaces allowing the host application to notify the runtime of events and/or to obtain additional information during host application process execution.

27. A computing device comprising the processor coupled to the computer-readable medium of claim 1.

28. A computing device for enhanced runtime hosting, the computing device comprising:

means for identifying, by a runtime one or more execution environment abstractions implemented by a host application, the host application for hosting the runtime;

during execution of runtime managed code and responsive to an action or event associated with an identified one of the respective execution environment abstractions, means for the runtime to interface with specific ones of the execution environment abstractions.

29. A computing device as recited in claim 28, wherein the execution environment abstractions correspond to memory management, threads/tasks, I/O completion, synchronization, runtime entry/exit notification, security context, impersonation, runtime configuration, executable service code abstractions, and/or garbage collection (GC).

30. A computing device as recited in claim 28, wherein the execution environment abstractions comprise means for interfacing with an object associated with the host application, the runtime interfacing with the object responsive to a specified event or criteria that occurs during host application execution.

31. A computing device as recited in claim 28, wherein the execution environment abstractions comprise means for the host application to configure operations of the runtime, notify the runtime of an event, and/or to obtain additional information during host application process execution.

32. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to request a memory allocation.

33. A computing device as recited in claim 28, wherein the execution environment abstractions comprise means for the runtime to:

communicate a low memory notification from the OS to the host application; and/or

inform the host application of consequences of failing a particular memory allocation.

34. A computing device as recited in claim 28, wherein the execution environment abstractions comprise means for the runtime to create a new thread/task.

35. A computing device as recited in claim 28, wherein the execution environment abstractions comprise means for the runtime to reuse or pool a runtime-implemented portion of a task.

36. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to notify the host application that a task cannot be moved to a different physical OS thread and cannot have execution of the task blocked for a specified window of time.

37. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to:

identify a runtime interface for the host application to invoke when a task has been moved to a runnable or non-runnable state; and

if the task has been moved to a non-runnable state, specify that the task is to be rescheduled by the host application.

38. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to:

queue a thread/task to a host application implemented thread pool;

set a size of the host application implemented thread pool; and/or

query the host application implemented thread pool.

39. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to:

provide a callback to the host application for notifying the runtime that a task has been moved to a different locale or a locale has changed for the task; and/or

notifying, by the runtime via the at least one specific interface or object, the host application, that a task has been moved to a different locale or a locale has changed for the task.

40. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to delay host application abort of a task.

41. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to adjust priority of a task associated with the host application.

42. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to notify the host application that a task/thread is to leave the runtime into unmanaged code.

43. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to notify the host application that a task/thread is to reverse-leave the runtime into unmanaged code.

44. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to notify the host application that a task/thread is to enter the runtime from unmanaged code.

45. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to notify the host application that a task/thread is to reverse-enter the runtime from unmanaged code.

46. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to indicate to the host application a maximum number of threads/tasks that will be available to service requests on one or more I/O completion ports.

47. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to bind a handle to an I/O completion port of the host application.

48. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to indicate a runtime implemented callback to the host application, the runtime implemented callback for calling by the host application when an asynchronous I/O operation completes.

49. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to supply a runtime implemented callback to the host application, the runtime implemented callback for invoking by the host application when an asynchronous I/O operation completes, the runtime implemented callback giving the host application an opportunity to communicate custom state information to the runtime implemented callback.

50. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to:

generate a task; and

create one or more synchronization objects for the task to substantially ensure host application knowledge of a lock on the task, the one or more synchronization objects comprising a critical section, a manual and/or auto-reset event, a semaphore, a reader/writer lock, and/or a task monitor.

51. A computing device as recited in claim 28, wherein the execution environment abstractions further comprise means for the runtime to notify the host application of one or more runtime interfaces exposed by the runtime, the runtime interfaces for the host application to notify the runtime of an event and/or to obtain additional information during process execution.